Part – 1

Objective: Use Spark Map Reduce functions in order to compare friends lists and create a new list of user pairs and their mutual friends

Data: The data consists of approximately **88,000 friend pairs**. Each line is a 'user-friend' and a single 'friend-from-friend-list', each represented by an id number. So, the data consists of many rows similar to '0 1' or '176 1130' with the first number being the 'user' and the second number being one of their friends.

However, I was unable to get this data format to work and had to create my own.

```
0 1 2 4 5 6 7
1 0 2 3 5 9
2 1 0 6 7 8 9
3 1 4 5 7 9
4 1 3 5 8
5 0 1 3 4
6 0 2 7 8 9
7 0 2 3 6 9
8 2 4 6
9 1 2 3 6 7
```

Evaluation: I had many problems with the assignment. Most I managed to solve, but the one I was left with was converting all the rows, '0 1' , '0 2' , '0 4' into a single row '(0) (1,2,4)'. In the end I was unable to solve this issue and instead made my own data file that reflects how the data should have looked after the first transformation.

From that point it was easy, I sorted the keys, then had to create a function to perform a true mathematical set union. Or taking two sets and creating a new set that consists of the items found in both sets. From there it was just another map reduce problem.

Conclusion: This problem was quite tricky, and provided some challenges that I thought would be simple at first, but I was unable to solve in the time I had. However, by pre-formatting the data in the proper way I was able to solve the problem still.

```
('0 1', {'5', '2'})
('0 2', {'7', '1', '6'})
('0 4', ['1', '2', '4', '5', '6', '7'])
('0 5', {'4', '1'})
('0 6', {'7', '2'})
('0 7', {'6', '2'})
('1 2', {'0', '9'})
('1 3', {'9', '5'})
('1 5', {'0', '3'})
('1 9', {'3', '2'})
('2 6', {'7', '0', '8', '9'})
('2 7', {'0', '9', '6'})
('2 8', {'6'})
('2 9', {'7', '1', '6'})
('3 4', {'1', '5'})
('3 5', {'4', '1'})
('3 7', {'9'})
('3 9', {'7', '1'})
('1 4', ['1', '3', '5', '8'])
('4 5', {'1', '3'})
('4 8', set())
('6 7', {'0', '9', '2'})
('6 8', {'2'})
('6 9', {'7', '2'})
('7 9', {'2', '6', '3'})
```

CS5542 - Big Data Apps and Analytics

LAB ASSIGNMENT #3

2. Spark Data Frames

Datasets:

1. **FIFA World Cup:**
   https://www.kaggle.com/abecklas/fifa-world-cup#WorldCupMatches.csv
2. **Kickstarter Projects**
   https://www.kaggle.com/kemical/kickstarter-projects
3. **Google-Landmarks Dataset**
   https://www.kaggle.com/google/google-landmarks-dataset

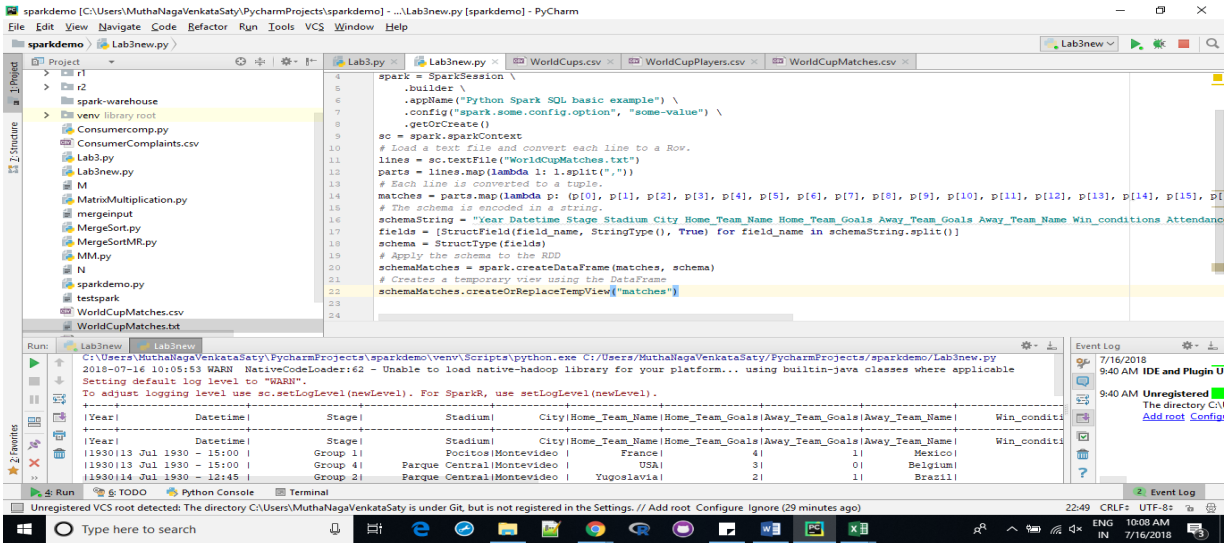
   a. **Create a Spark DataFrame using one of datasets, trying to use all different StructType.**
   b. **Perform 10 intuitive questions in Dataset (e.g.: pattern recognition, topic discussion, most important terms, etc.). Use your innovation to think out of box.**
   c. **Perform any 5 queries in Spark RDD's and Spark Data Frames. Compare the results**

Data Set Considered: FIFA World Cup

Steps:

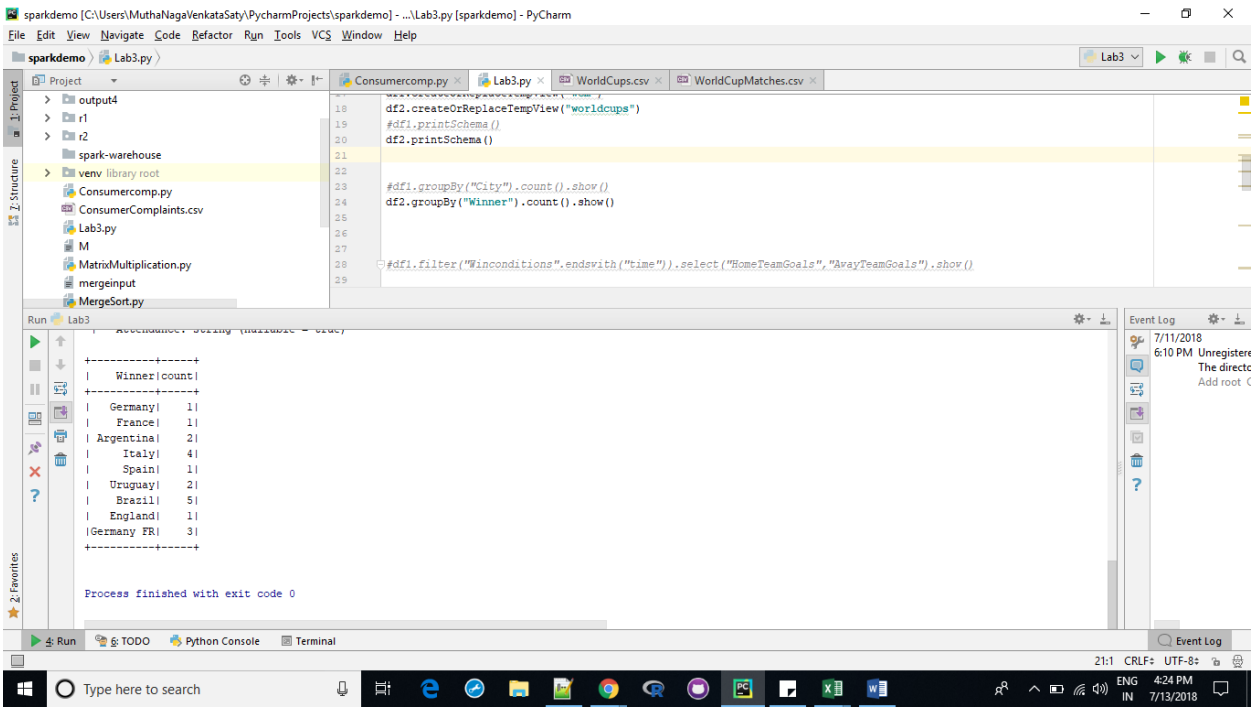
1. Creating a Spark DataFrame using different Struct type.
2. Load the data through a csv file.
3. Querying on either Data frame directly or through SQL.

Step 1: Create Spark DataFrame and Loading CSV File

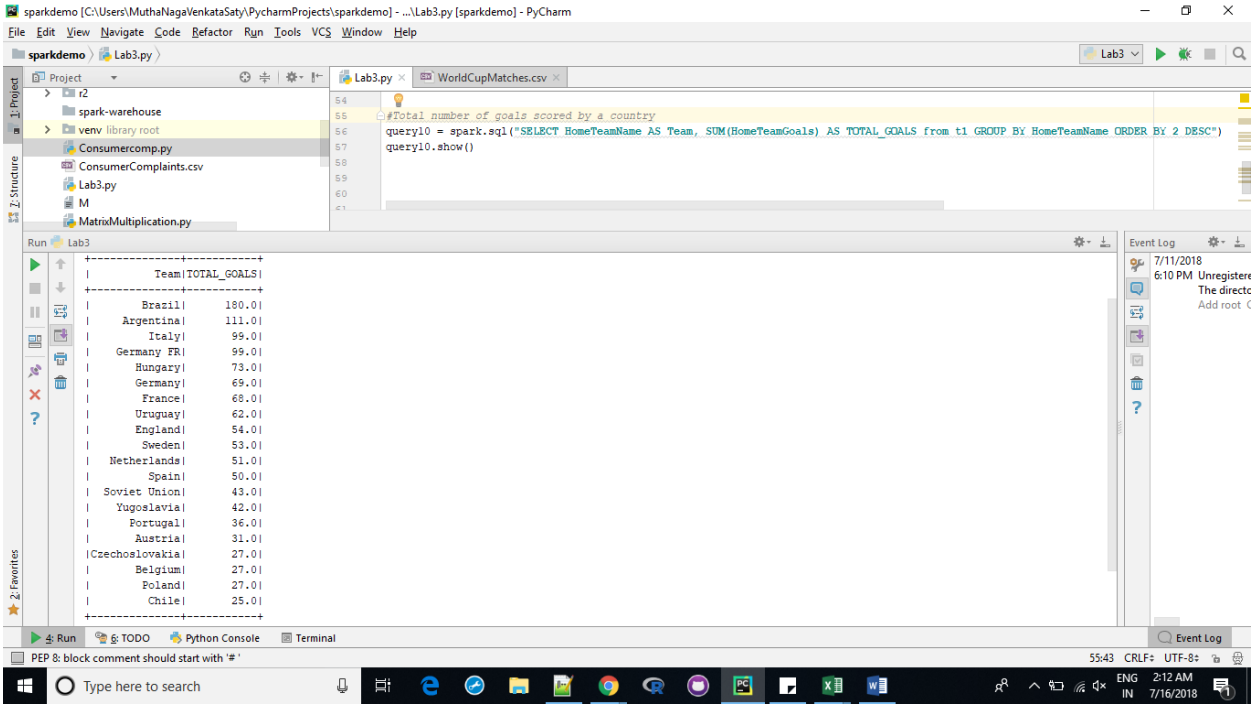


Queries:

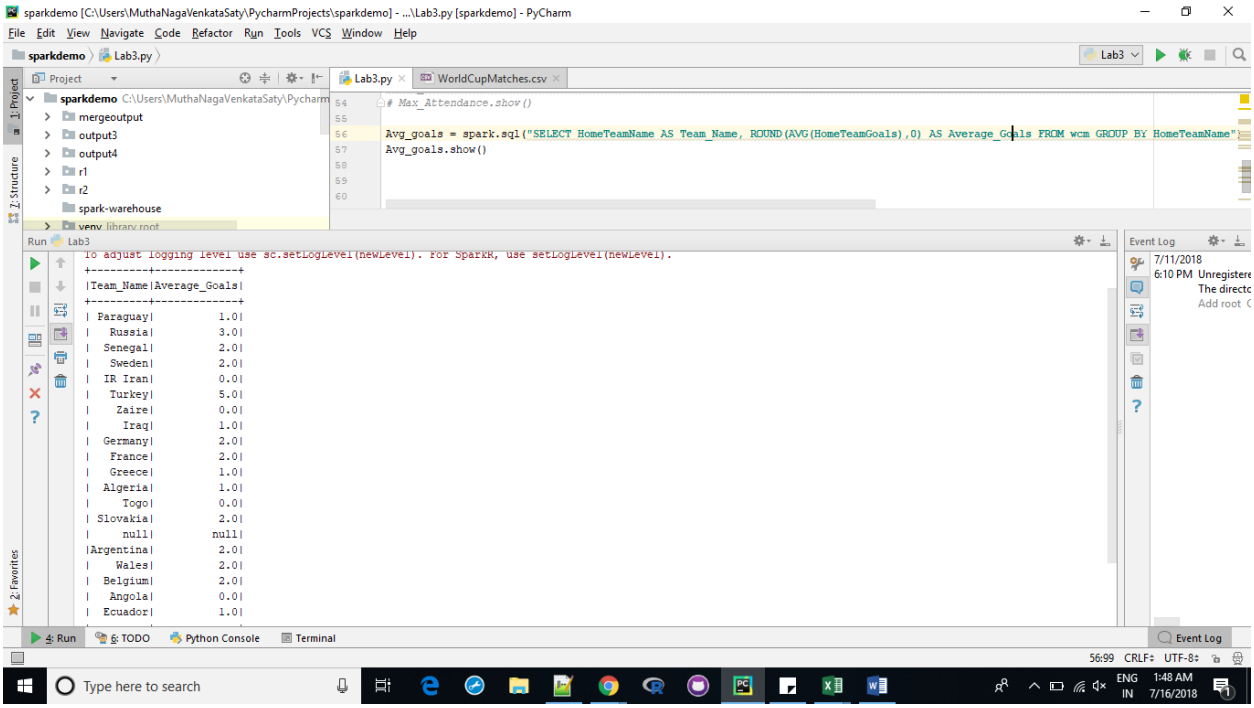
#Query 1: This query is directly performed on data frame itself. This query lets us know how many times each team has won the world cup till 2014.



#Query 2: This query gives the total number of goals scored by each team with the team scoring more goals as the first.

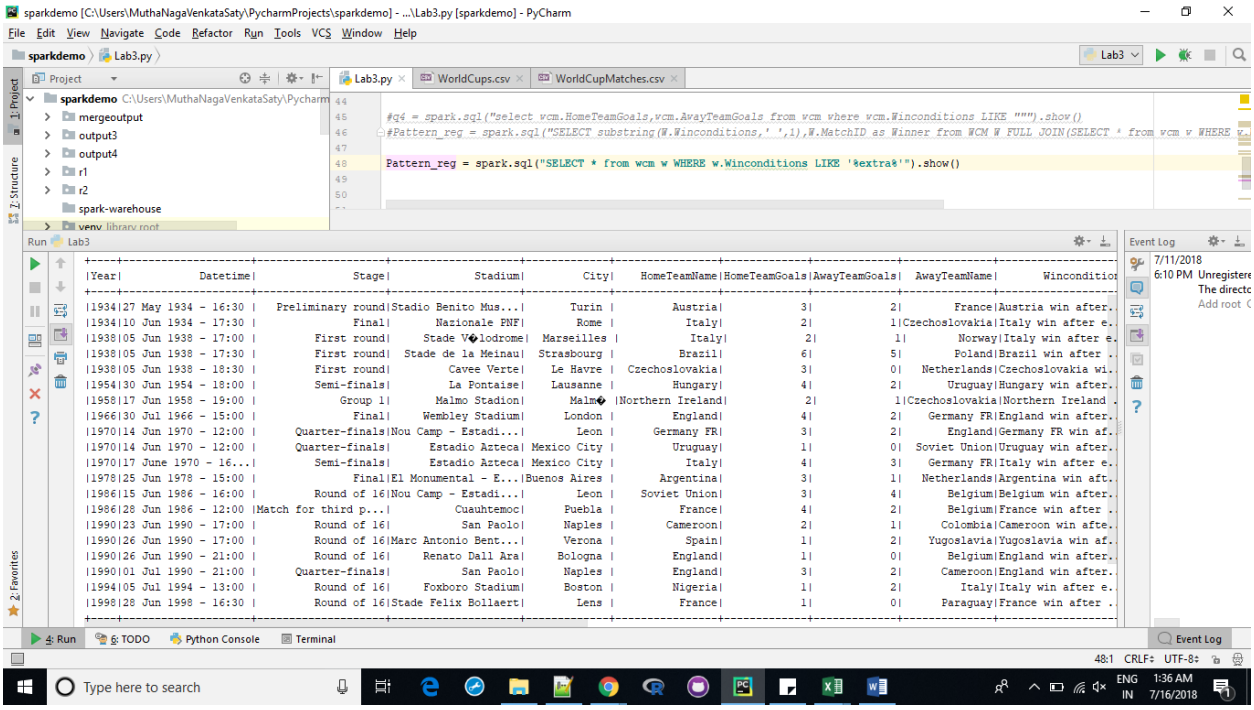


#Query3: This query tells the average number of goals scored by team.

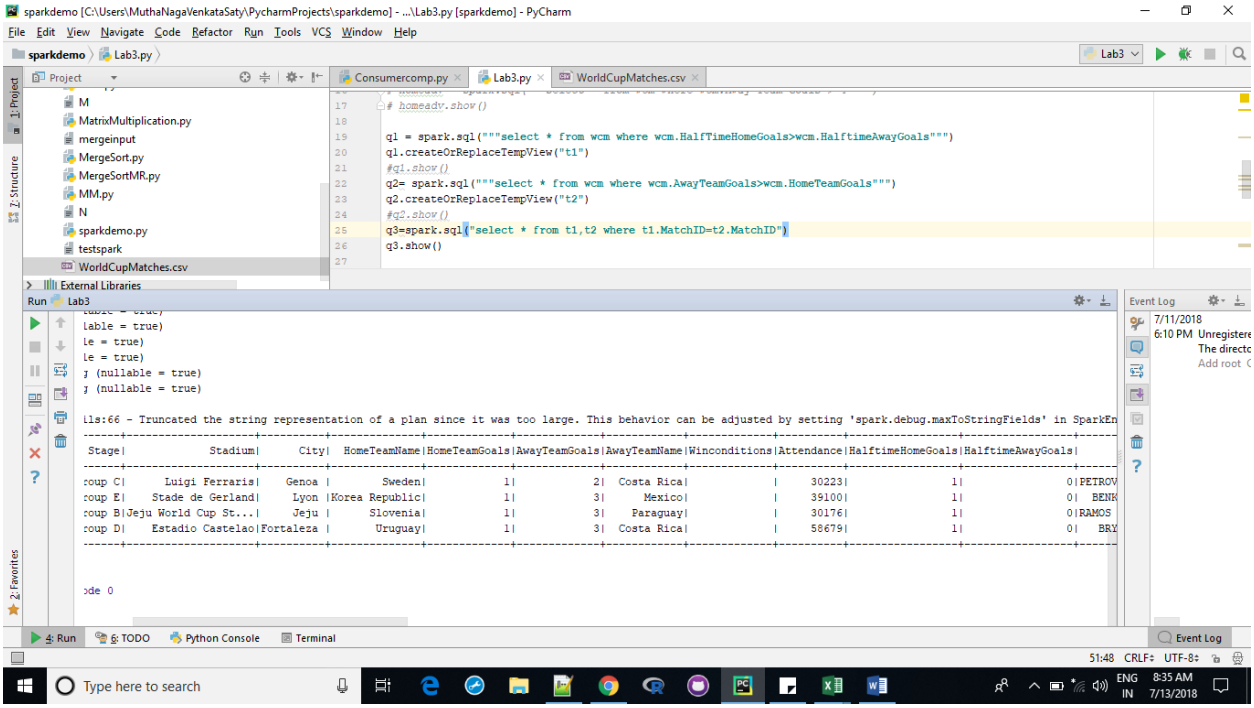


#Query 4:

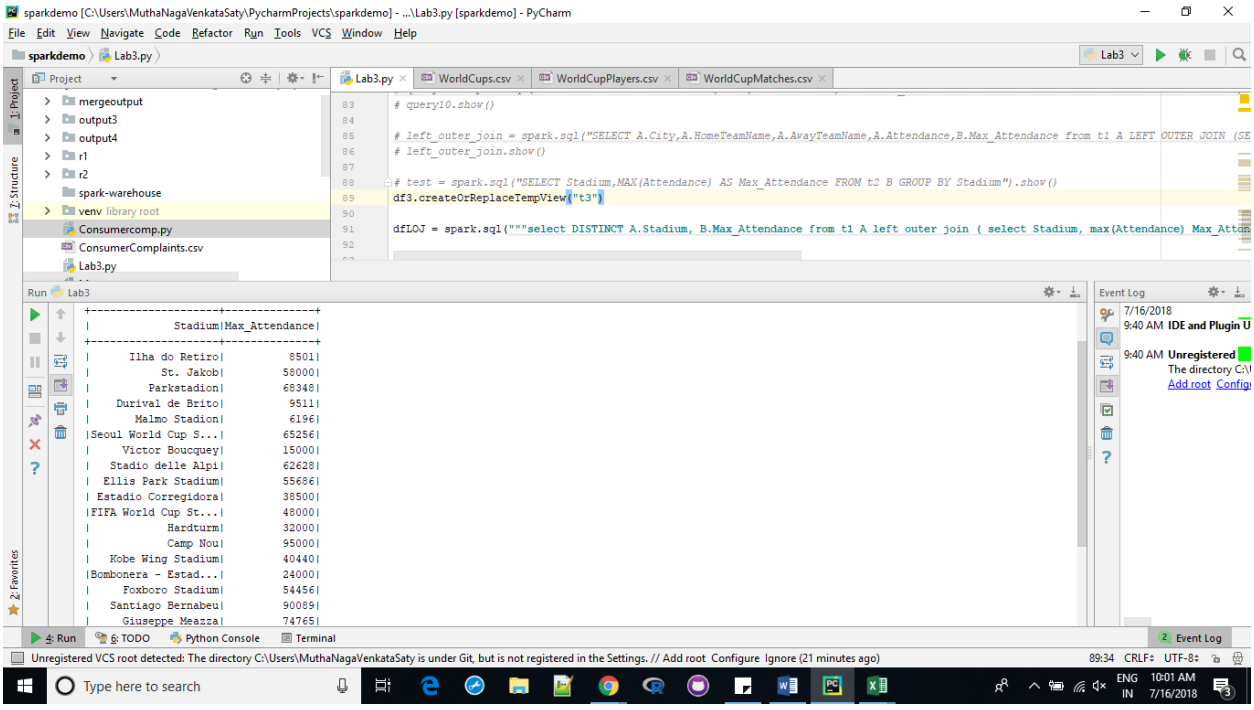
This query is bit about the pattern recognition. I found the matches where the matches have taken extra time to win the game.



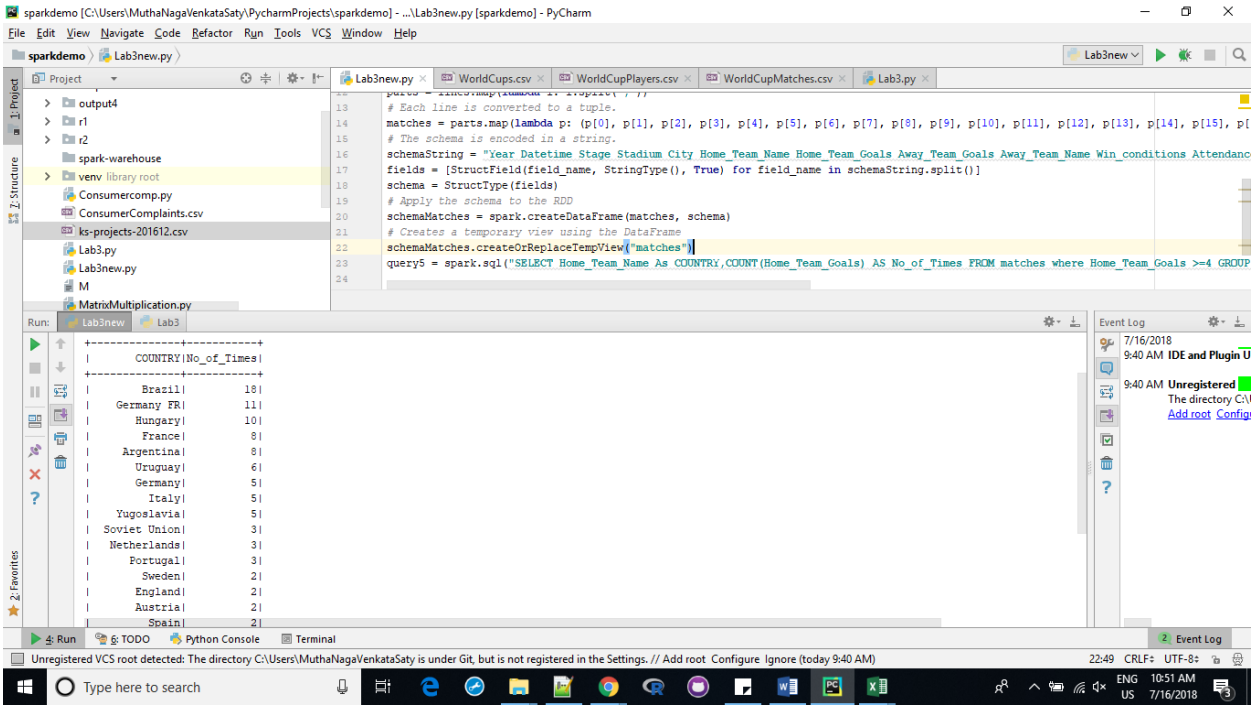
#Query 5: In this query, I tried to find where the Home Team leads the Away Team by halftime but the Away Team wins the match.



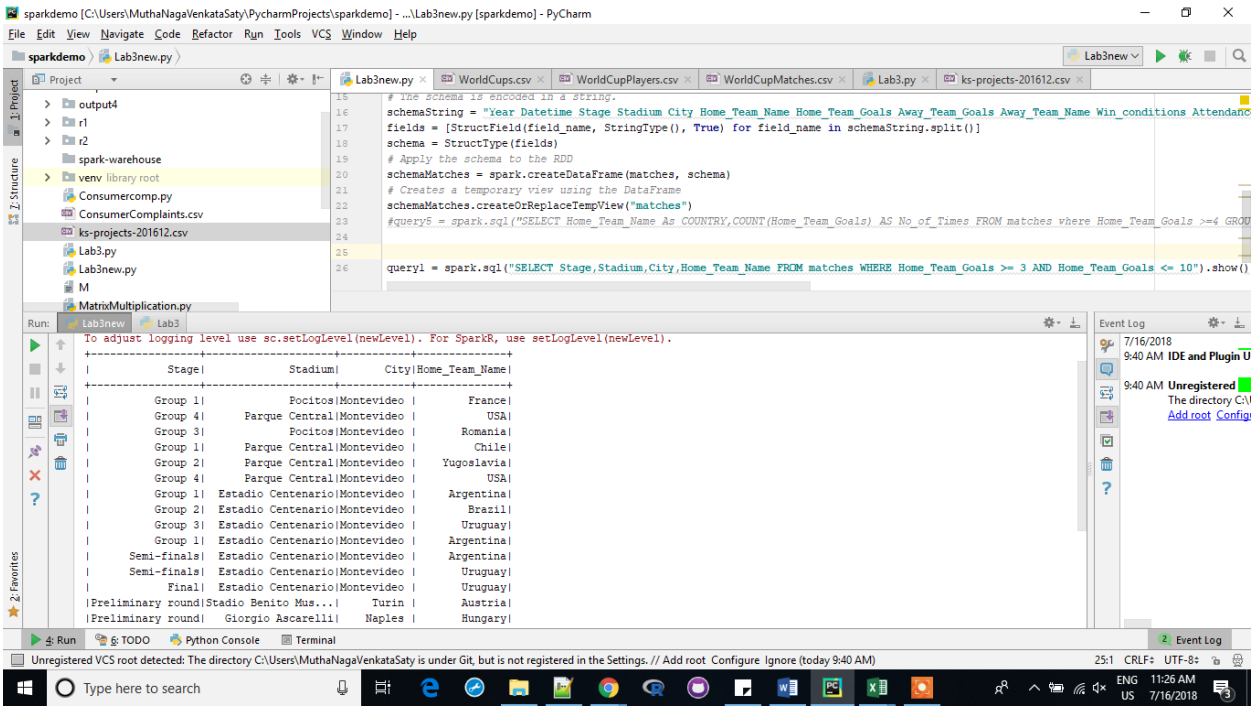
#Query 6: In this query, I tried to find the max attendance for a stadium



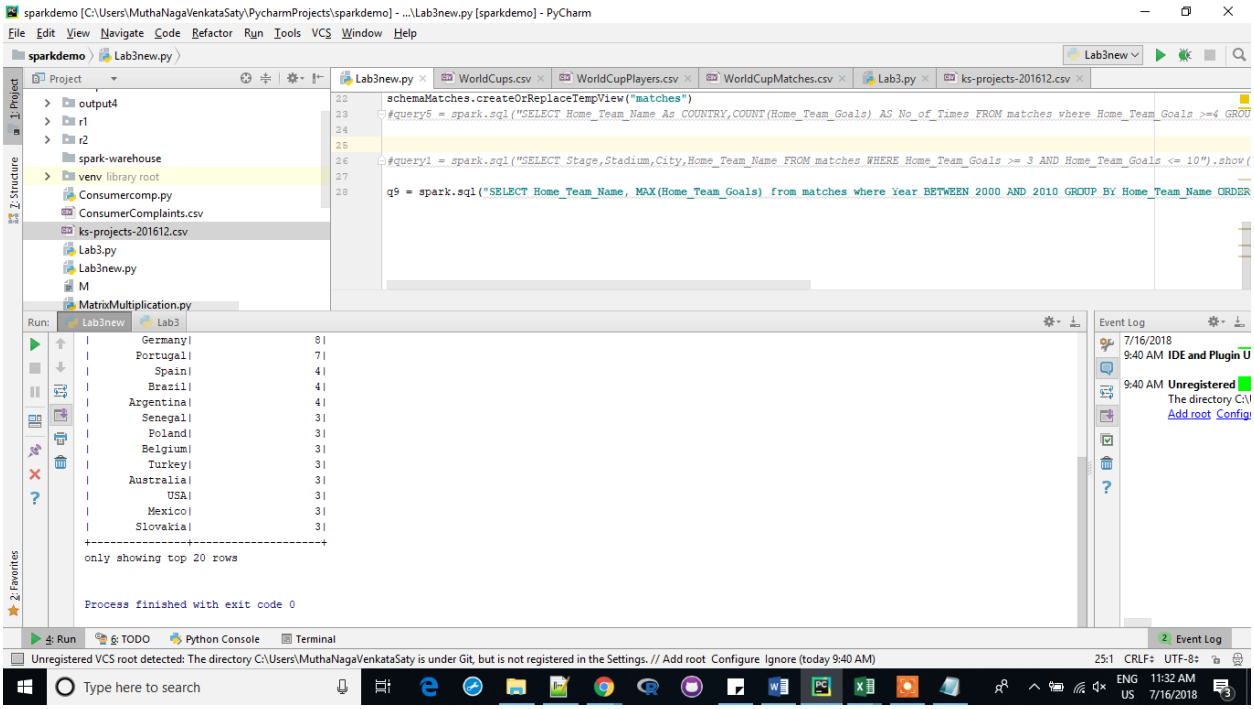
#Query 7: I tried find the teams who have scored more than 4 goals till date using the Built In functions



#Query 8: In this query I would like to know the goals between a particular range.



#Query 9: I tried to find the max goals between a range of years



#Query 10: I tried to find the teams and the years they won using **collect_set** function